Middlebury

CSCI 146: Intensive Introduction to Computing

Fall 2025

Lecture 1: Introduction

Goals for today

- Introduce ourselves.
- Introduce the course:
    - What will we be doing and how?
    - What will you know at the end of the semester?
- Define algorithm, semantics, and syntax.
- Introduce Picobot and implement a Picobot algorithm.
- Evaluate arithmetic expressions over integers and floats.
- Describe the concept of "type".

Visit go/cs146, click on **Reactions** tab, then click one of the emojis.
Use this any time! (during class)

Introductions!

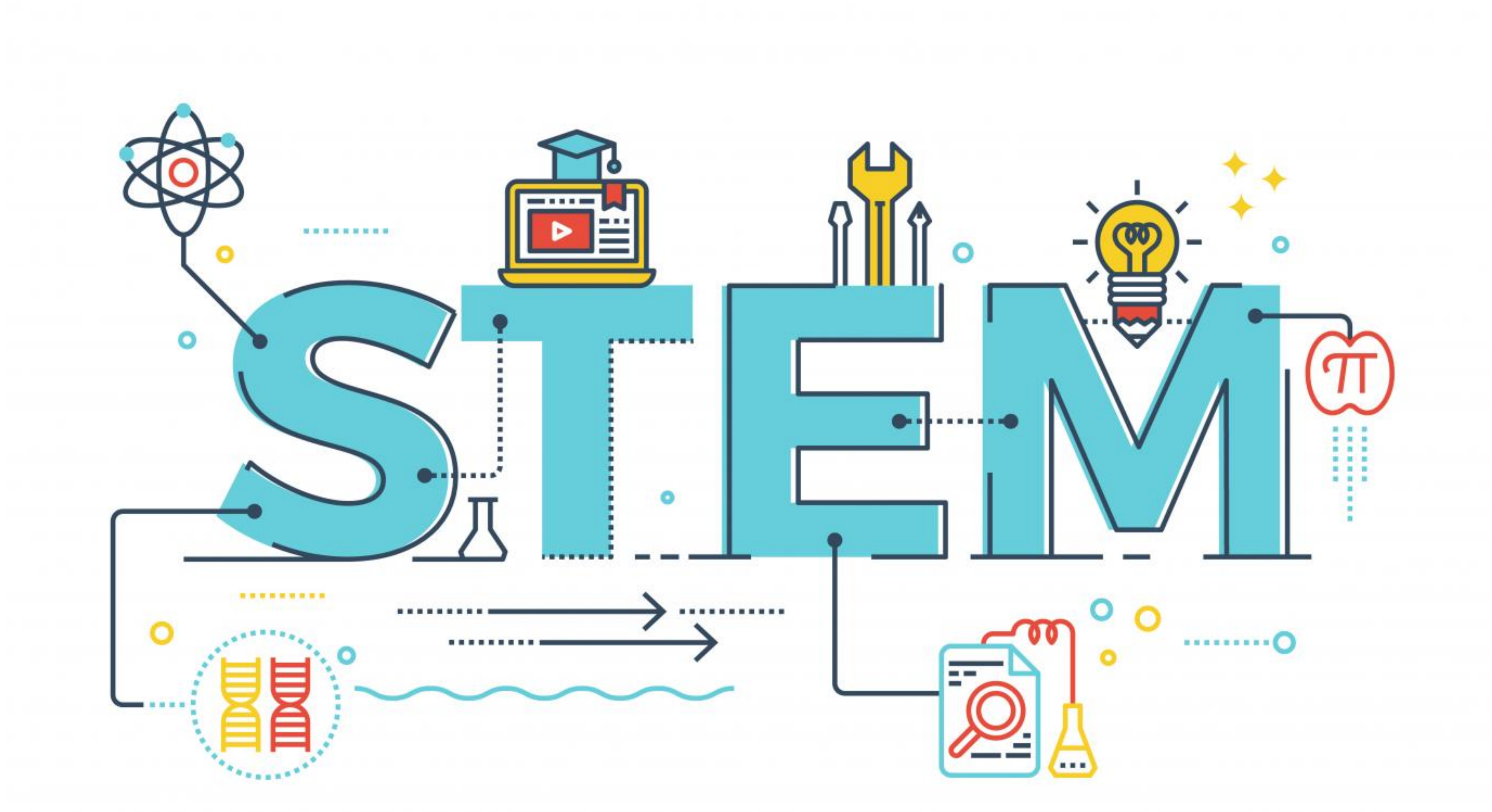
In groups of 2 - 3:

- Introduce yourselves!
- If you're new to Middlebury, what's something cool you have explored so far?
- If you've been at Middlebury for a while, what's something you recommend doing?
- What brings you to CSCI 146? What is computer science?

"Algorithmic thinking/problem solving usually (but not always) involving computers.”

What is the difference between CSCI145 and CSCI146?

- Mostly pace, with 1 - 2 extra weeks of content.
- Do you need to have programmed before? No.
- This course is for those with a background in math or science (e.g. having taken college-level math/science courses) OR for those with *some* programming experience.



A little about me...

- Please call me Philip.
- My research involves developing geometry tools for fluid simulations.
    - How can we make fluid simulations on the Earth faster (and accurate)?
    - How can we estimate (quickly, accurately) aircraft performance using a computer?
- I also teach computer graphics (techniques used in animation!).

Fluid simulation on a sphe...

Wall-modeled Large Eddy Lattice Boltzmann Simula...

(Rightmost video from the `openlb` group)

For me, computer science helps solve physics problems (specifically fluid flow) that are impractical (or impossible) to solve by hand.

What will we do in this class?

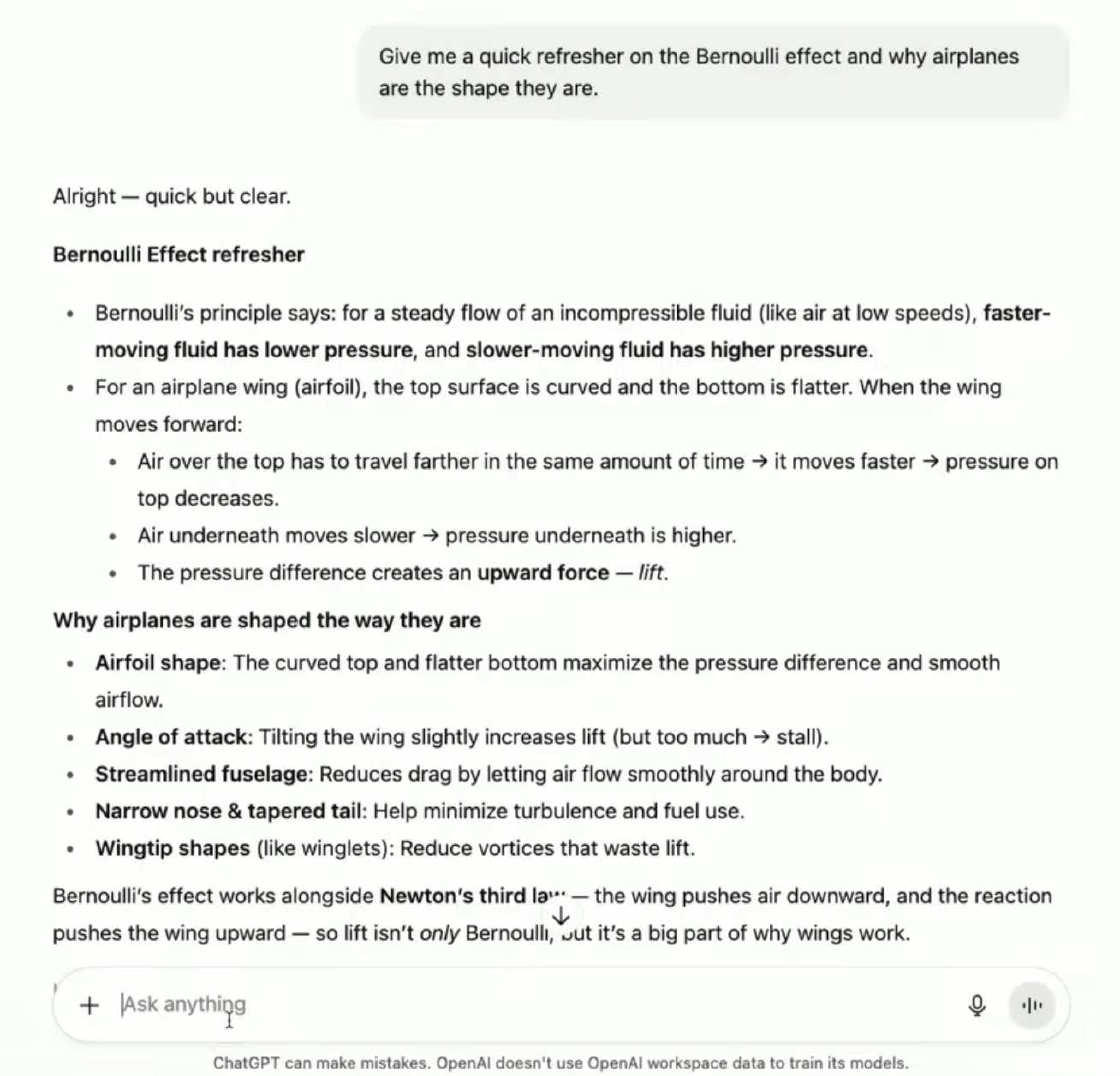
- **Lectures** on Mondays and Wednesdays. Please bring a computer for programming exercises.
- **Readings** posted before class so you can follow along. Annotated slides posted after class.
- Optional practice PrairieLearn problems posted early in the week.
- We will use a mix of **EMRN** (Exemplary, Meets expectations, Revision needed, Not assessable) and **Complete/Incomplete** for grading.
  - Final grade determined by "bundles" (see syllabus for more details).
- **Quizzes** and **Labs** on Fridays.
  - Quizzes graded based on *topics* achieved in EMRN model. Can retake topic questions at later dates.
  - Labs graded Complete/Incomplete (all labs due at the end of the semester).
- **Programming Assignments** (released on Fridays):
  - Graded using EMRN model.
  - Initial and Final due dates so you can revise your work!
- Two programming **projects** (kind of like take-home programming tests). Graded using EMRN.
- **Two midterms** (Thursday October 9th and Thursday November 13th) and a **final exam** (TBD):
  - Like quizzes, exams are graded based on topics (using EMRN) with an opportunity for retaking topic questions.
- I want you to come to class! Final grade modifiers (+/-) will be used based on attendance records.
  - Need to miss class? Submit this form for an excused absence.
  - If you miss more than 1 consecutive week of classes, then I will reach out to check in (copying your dean and advisor).
  - There will be no Zoom link for class.

I would suggest bookmarking go/cs146.

A note about using AI in CSCI 146.

- I recognize that it's a useful tool.
- But it's not perfect. These tools make mistakes and may be misleading.
- We need to know when, and understand why.
- Ultimately, we want you to build a strong foundation in CS.
- I suggest thinking of it as an **assistant to help you learn**, but **not a replacement to do the work for you**.

Introducing GPT-5



Give me a quick refresher on the Bernoulli effect and why airplanes are the shape they are.

Alright — quick but clear.

Bernoulli Effect refresher

- Bernoulli's principle says: for a steady flow of an incompressible fluid (like air at low speeds), **faster-moving fluid has lower pressure**, and **slower-moving fluid has higher pressure**.
- For an airplane wing (airfoil), the top surface is curved and the bottom is flatter. When the wing moves forward:
  - Air over the top has to travel farther in the same amount of time → it moves faster → pressure on top decreases.
  - Air underneath moves slower → pressure underneath is higher.
  - The pressure difference creates an **upward force** — *lift*.

Why airplanes are shaped the way they are

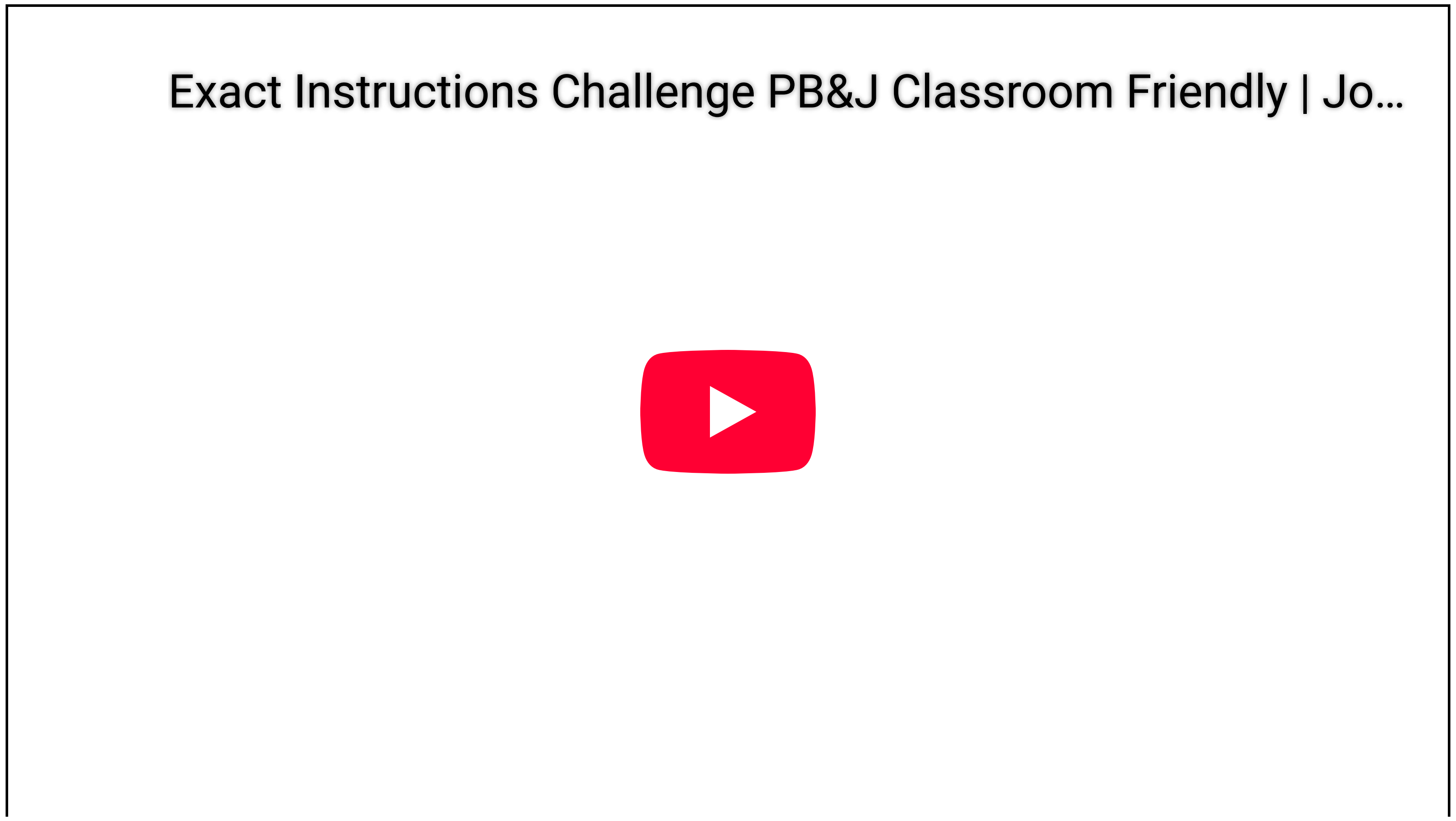
- **Airfoil shape**: The curved top and flatter bottom maximize the pressure difference and smooth airflow.
- **Angle of attack**: Tilting the wing slightly increases lift (but too much → stall).
- **Streamlined fuselage**: Reduces drag by letting air flow smoothly around the body.
- **Narrow nose & tapered tail**: Help minimize turbulence and fuel use.
- **Wingtip shapes** (like winglets): Reduce vortices that waste lift.

Bernoulli's effect works alongside **Newton's third law** — the wing pushes air downward, and the reaction pushes the wing upward — so lift isn't *only* Bernoulli, but it's a big part of why wings work.

ChatGPT can make mistakes. OpenAI doesn't use OpenAI workspace data to train its models.

Let's talk about algorithms!

Proposition: an algorithm is *like* a recipe.



Exact Instructions Challenge PB&J Classroom Friendly | Jo...

But the recipe analogy is imperfect. More formally, an algorithm:

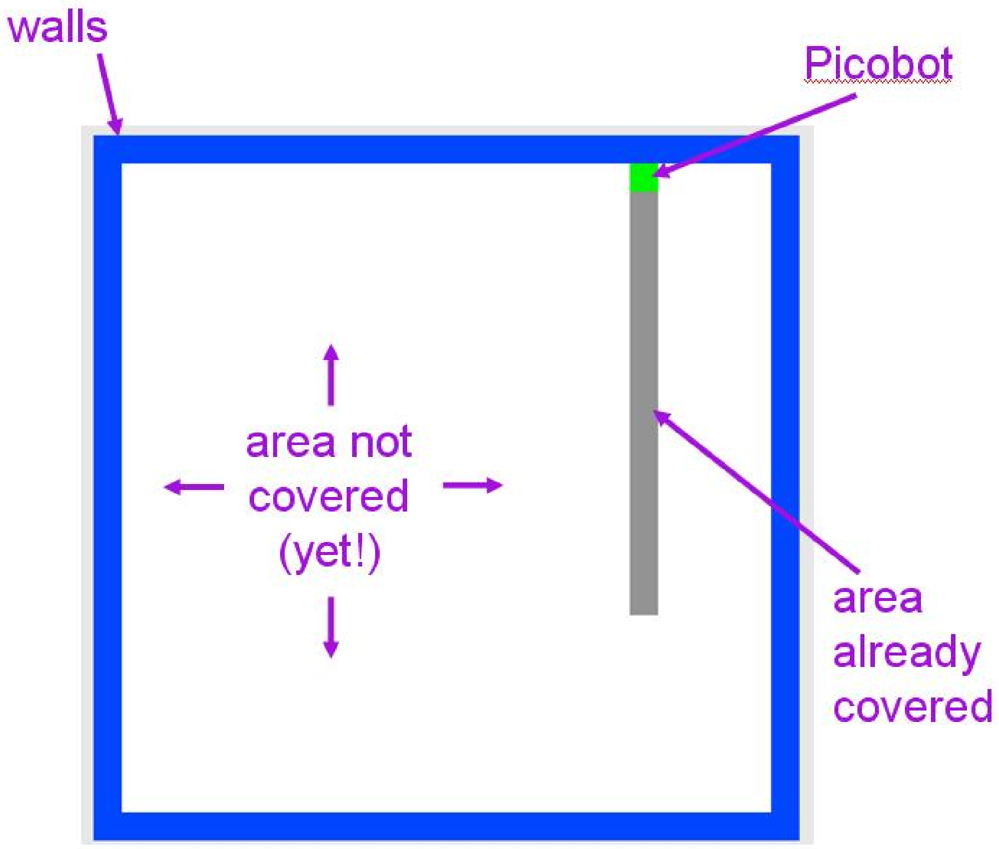
- Has a finite number of instructions or steps.
- Each instruction (step) is well-defined (i.e., not ambiguous) and computable.
- Eventually halts (i.e., executes in a finite amount of time).
- Solves a general class of problems.

Recall that we want to use a computer to solve problems.

- We need to define problems inputs and outputs.
- We need to develop an algorithm to solve the problem.
- We need a way to *implement* our algorithm.

Using `Picobot` as an example (visit: https://www.cs.hmc.edu/picobot/):

- **Goal:** visit every square in an empty room.
- **Constraints:** walls, can only move in cardinal directions (NEWS).
- **Algorithm:** let's do a 2-minute brainstorming in groups of 2-3.
- **Implementation:** `STATE SURROUNDINGS -> MoveDirection NewState`

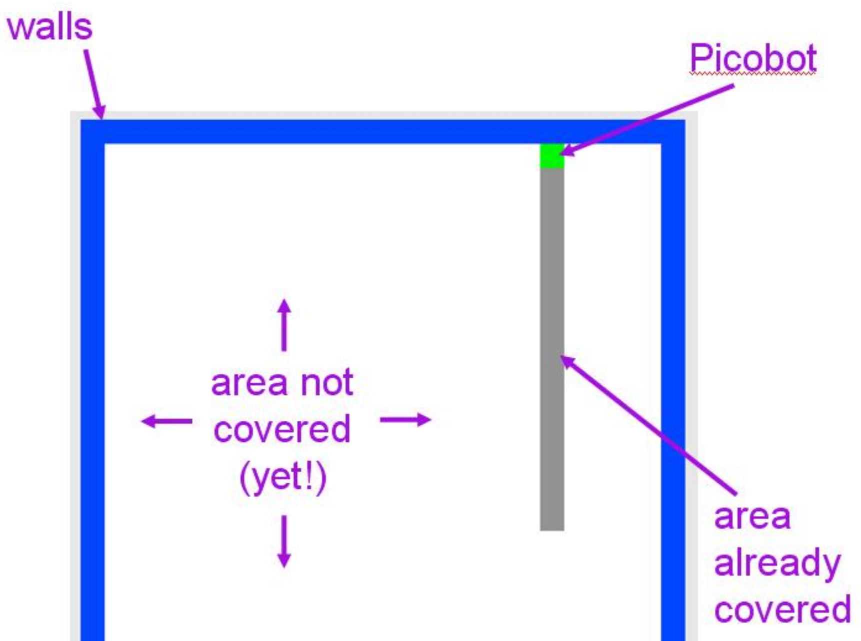


Goal: whole-environment coverage with only *local sensing*…

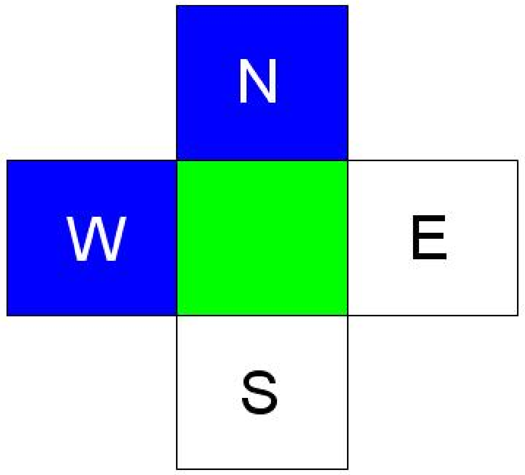
Picobot rules (examples).

`STATE SURROUNDINGS -> MoveDirection NewState`

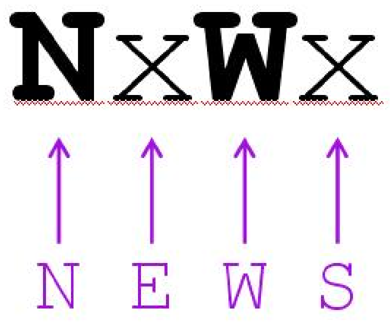
```
# "If Picobot is in state 0 and senses xxxS (only a wall to the south)
# it should move north and stay in state 0”
0   xxxS   ->   N   0

# “If Picobot is in state 0 and senses nothing to the north and
# anything (* wildcard) to the east, west, and south (wall or not)
# it should move north and stay in state 0”
0   x***   ->   N   0
```



Goal: whole-environment coverage with only ***local sensing***…



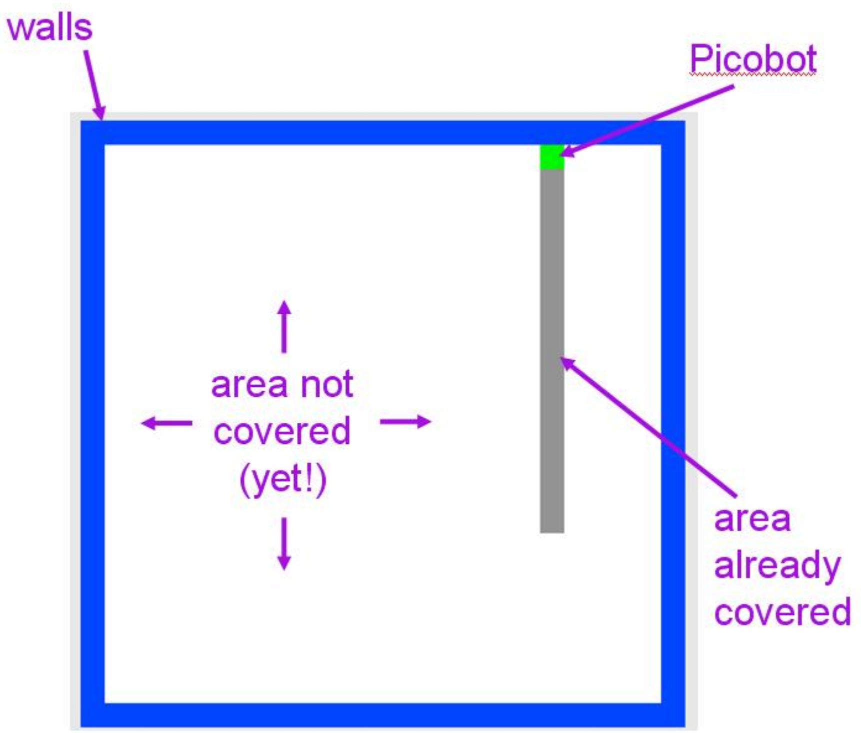
Here, picobot’s surroundings are



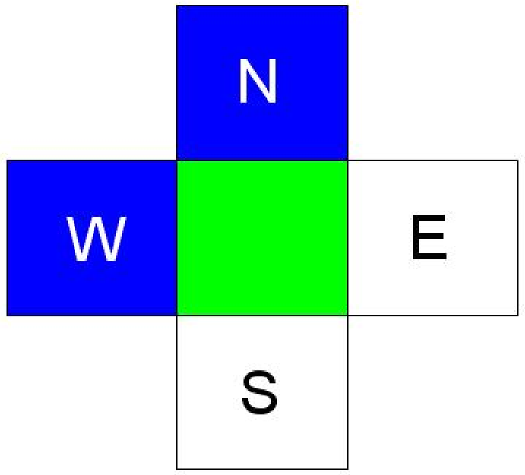
Surroundings are always in NEWS order.

Will `Picobot` visit the entire room with this? A: Yes, B: No.

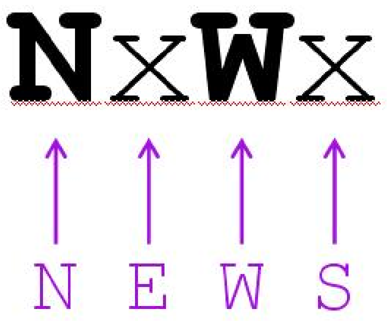
```
#
# Hashtag lines are optional comments that are ignored by Picobot
#

# State 0 with nothing N: Go one step N
0 x*** -> N 0

# State 0 with a wall to the N: Go W and into state 1
# ** This will crash if picobot has a wall to the W! **
0 N*** -> W 1

# State 1 with nothing to the S: Go one step S
1 ***x -> S 1

# State 1 with a wall to to the S: Stay put and go into state 0
1 ***S -> X 0
```



Goal: whole-environment coverage with only ***local sensing…***



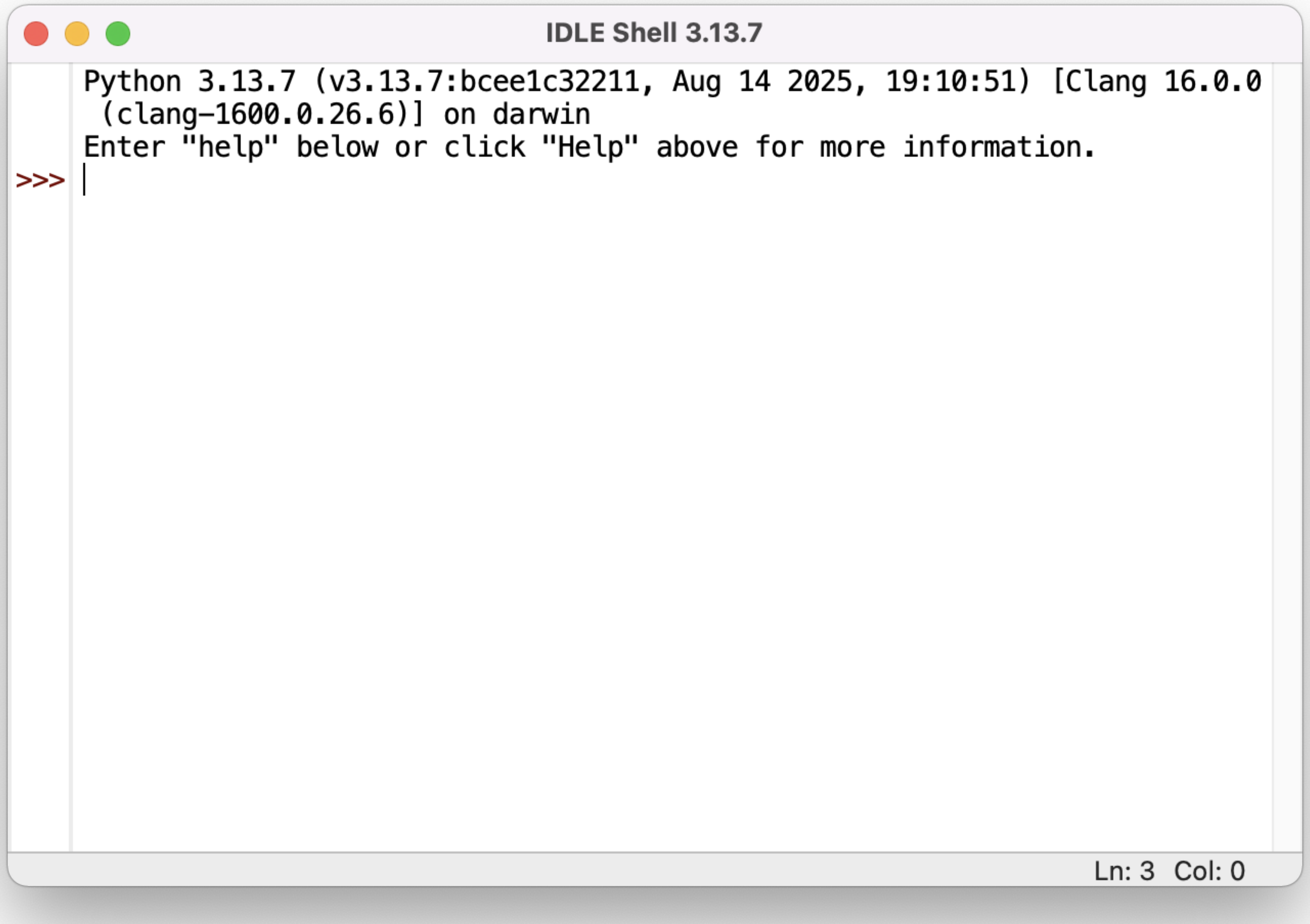
Here, picobot's surroundings are



Surroundings are always in NEWS order.

Let's get started with `Python`!

- Visit https://www.python.org/downloads/ and download `Python 3.13`.
- Start the installer. If on Windows, make sure to check the box for **Add python.exe to PATH**.
- This will install `Python` as well as something called `IDLE` (we will only use `IDLE` today to get started). Open `IDLE`.
- We'll now type some commands in the "shell" together (examples in the readings).



```
Python 3.13.7 (v3.13.7:bcee1c32211, Aug 14 2025, 19:10:51) [Clang 16.0.0
 (clang-1600.0.26.6)] on darwin
Enter "help" below or click "Help" above for more information.
>>> 
```

Which of the following gives a result of `16` in `Python`?

A. `8 // 2 * 4`

B. `8 / (2 * 2 + 2)`

C. `8 // (2 * 4)`

D. None of the above.

One final note: please come to class.

The word COMPUTER means...

ONE FINAL NOTE: PLEASE COME TO CLASS.

The word COMPUTER means...

"together"

similar to the word "community"

Summary

- The shell is good to test a few small expressions, *scripting* is better for complete programs (next class).
- Be careful with types.
- **Tips for succeeding in this course:**
    - Come to class!
    - Go through the readings and notes some time after class.
    - Work on the PrairieLearn problems throughout the week.
    - Start the Programming Assignments and Projects early.
    - **Come to office hours!**
      There will likely be juniors/seniors working on Computer Graphics problems too so you can chat with them about CS.
- **Before next class:**
    - Complete the Introduction Form so I can get to know you :)
    - Go through the steps in the first reading to install `VS Code` and get set up with `Python`.
    - Click on Practice Problems in the **Calendar** and log into the PrairieLearn system.
- (For fun and practice with Gradescope): use the `empty_room.txt` file linked in the first reading to have Picobot traverse the entire room and submit to Gradescope (Programming Assignment 0).